Combinatorial Approach to Key Generation using Multiple Key Spaces for Wireless Sensor Networks

K G Srinivasa[#1], Poornima V[#2], Archana V[#3], Reshma C[#4], Venugopal K R[*5], L M Patnaik[+6]

[#]*Data Mining Laboratory, Department of Computer Science and Engineering, M S Ramaiah Institute of Technology, Bangalore.*
[1] reshmachannappa23@gmail.com
[2] poornibharadwaj@gmail.com
[3] archana.venkatesha@yahoo.co.in
[4] srinivasa.kg@gmail.com

[*]*University Visvesvaraya College of Engineering, Bangalore University, Bangalore.*
[+]*Defence Institute of Advanced Technology, Pune.*

***Abstract*—Key generation is an important challenge among the issues related to security in wireless sensor network. Each node has a set of keys called key-chain rather than a single shared key. The key-chains are generated at the base station and stored in the ROMs of sensor nodes prior to deployment. This paper brings in the idea of deterministic approach to key generation. There is no need to assume a probability that any two nodes are neighbors. We use combinatorics based approach using a block design technique called symmetric Balanced Incomplete Block Design (BIBD). We construct multiple key-spaces out of a key pool from which the key-chains are formed. The main contribution to this work is the use of multiple key spaces to decrease memory utilization at each sensor node, retain connectivity and still not hamper the resilience of the network. It decreases the redundancy in key generation. It eliminates the dependency between the number of keys in the key-chain and number of nodes in the network.**

I. Introduction

Wireless sensor networks consist of many tiny sensor nodes deployed at a high density over region requiring surveillance and monitoring. The sensor nodes typically consist of one or more sensing elements, battery, low powered radio transmitter/receiver, microprocessor and limited memory. Sensor networks deployed in a hostile environment are prone to malicious attacks like eavesdropping, masquerading, traffic analysis etc. Hence, security is more important in sensor networks than in traditional networks.

To provide security, communication between nodes should be encrypted and authenticated. Encryption and authentication algorithms require that the communicating nodes share a secret key. The problem here is how to set up secret keys between communicating nodes? This problem is known as the *key agreement* problem. There are basically three types of general key agreement schemes: trusted server scheme, self-enforcing scheme and key pre-distribution scheme. The trusted server scheme depends on a trusted server for key agreement between nodes. E.g. Kerberos [1]. There is no trusted infrastructure in sensor networks. Hence this scheme is not suitable. The self-enforcing scheme depends on asymmetric cryptography, such as key agreement using public key certificates. However, limited computation and energy resources of sensor nodes often make it undesirable to use public key algorithms, such as RSA[2] or Diffie-Hellman key agreement [3]. The third type of key agreement scheme is key pre-distribution which is nothing but symmetric key cryptography. This scheme is most widely used and suitable for sensor networks.

The secret keys are stored in the ROMs of sensor nodes prior to deployment. The keys stored must be carefully selected so as to increase the probability that two neighboring nodes have atleast one key in common. Nodes that do not share a key directly must be able to establish a key path where each pair of nodes in the path shares a key.

There are a number of approaches to key pre-distribution. One naive approach is to store a master key in each of the nodes. The nodes use this master key to obtain a new pairwise key. Though it is simple to implement, it has a major disadvantage of single point of failure. If an intruder compromises a node, he is in possession of the master key with which he can compromise the whole network. Hence, this scheme is not resilient. Another approach is to store a set of $N-1$ keys in each node where one key is known only to one of the $N-1$ nodes (Assuming there are N nodes in the network). This scheme overcomes the resilience problem discussed in the previous scheme, since the capture of a single node will not reveal the secret keys between any pair of nodes. However, if the number of nodes increases, the set of keys to be stored in each node increases proportionally. Since the nodes have limited memory this scheme becomes impractical because N could be large. Moreover, if there are more nodes to be added to the network, the set of keys stored in the nodes have to be updated to include the keys for the new nodes.

Eschenauer *et al.*[4] proposed a random key pre-distribution scheme. In this scheme hundreds of keys are picked from a key pool and are preloaded in the sensor nodes. The keys along with their identities form the key-chains. The nodes exchange their key identities. They propose to employ Merkle

puzzle [5] to secure the key identities. However, the processing at each node increases. They show that for a key pool size of 10000, a key chain of length 75 is enough to obtain connection probability of 0.5. An enhancement to this scheme was proposed by Chan et al.[6]. In this scheme q-common keys are required instead of 1 to be able to increase the security of the communication. However this scheme requires larger key chains and smaller key pool compared to the previous scheme.

Random-pairwise key scheme is a modification over pairwise key scheme. Each node stores a random set of np pairwise keys instead of $n-1$ where n is the number of sensor nodes in the network and p is the probability that two nodes are connected. Blom proposed a scheme that allows any pair of nodes to find a pairwise secret key between them. Compared to $(N-1)$-pairwise key distribution scheme, Blom's scheme uses $\lambda+1$ memory space where λ is much smaller than N. Unlike $(N-1)$-pairwise key distribution scheme, Blom's scheme is not perfectly resilient against node capture. Instead, it has λ secure property: as long as an adversary captures less than or equal to λ nodes; uncompromised nodes are perfectly secure; when an adversary captures more than λ nodes all the nodes in the network are compromised. Blom's scheme uses one key pool to pick the keys. An extension to this scheme was proposed by Varshney *et al.* in [7] where multiple key spaces are incorporated.

One disadvantage of these random approaches is that the probability of success is low. Lee and Stinson[8] and Campete and Yener [9] proposed deterministic approach to key distribution using combinatorial design techniques to allocate keys to nodes such that the probability of key share between any two nodes is 1. The amount of memory required per node is some fractional power of the overall network size. The drawback of this scheme is that the same keys are shared between many pair of nodes leading to weaker resilience to node capture.

The rest of the paper is organised as follows. Section II gives a background of few of the topics required for understanding this paper. Section III gives the mapping between symmetric design and key distribution and also highlights the construction. Section IV gives the analysis and section V gives some results comparing the proposed symmetric design with multiple key-spaces and existing symmetric design with single key-space.

A. Our contribution

We use deterministic approach for key generation using a block design technique called symmetric *Balanced Incomplete Block Design* (BIBD). The main contribution to this work is the use of multiple key spaces to decrease the memory utilization and still retain connectivity without hampering the resilience of the network.

II. Background

The definitions of some of the terms used in this paper are listed below:

A. Key pool

A pool from which keys are picked is called a key pool. It is analogous to the universal set from which sets are formed.
Example 1: Let key pool, P = {1,2,3,4,5,6,7,8,9,10,11,12,13}

B. Key-space

A subset of the key pool which is used to construct key-chains for the sensor nodes is called as a key-space.
We know that 2^n subsets exist for a set containing n elements. Hence, for a key pool containing n elements 2^n key-spaces can be constructed.
Example 2: For the key pool P having 13 elements, 2^{13} key-spaces can be constructed. Each Key space is constructed using some criteria. Lets say, the key-spaces, KS_1 = {1,2,3,4,5,6,7}, KS_2 = {1,3,5,7,9,11,13} and KS_3 = {8,9,10,11,12,13} are 3 key-spaces used by a designer.

C. Key-chain

A set of keys assigned to each sensor node is called as key-chain. Key-chains are subsets of key-spaces.
Example 3: Two key-chains formed from KS_1 are k_1 = {1,2,3}, k_2 = {1,4,5} and are assigned to sensor nodes numbered N_1, N_2. k_3 = {1,7,9}, k_4 = {9,11,13} and k_5 = {3,5,7} are key-chains formed from KS_2 and are assigned to sensor nodes N_3, N_4, N_5.

D. Key-path

Sensor nodes that do not share a key directly may use a path where each pair of nodes on the path shares a key. Such a path is called as a key-path.
Example 4: If sensor nodes N_1 and N_2 has to communicate, they find 1 as the common key in their key-chains. Hence they use 1 as the secret key for their communication. Suppose N_1 and N_4 has to communicate, there is no common key in their key-chains. In such a case we say that there exists a key-path from N_1 to N_4 via N_3. This is because N_1 and N_3 share 1 as common key and N_3 and N_4 share 9 as common key. The network formed with these five nodes is as shown in figure 1.

E. BIBD

A *Balanced Incomplete Block Design* (BIBD) [10] is an arrangement of v distinct objects into b blocks such that each block contains exactly k distinct objects, each object occurs in exactly r different blocks, and every pair of distinct objects occurs together in exactly λ blocks. The design can be expressed as (v, k, λ), or equivalently (v, b, r, k, λ), where:

$$\lambda\ (v-1) = r(k-1) \text{ and } bk = vr.$$

A BIBD is called *Symmetric* BIBD or *Symmetric Design* when $b = v$ and therefore $r = k$. A Symmetric Design has four properties: every block contains $k = r$ elements, every

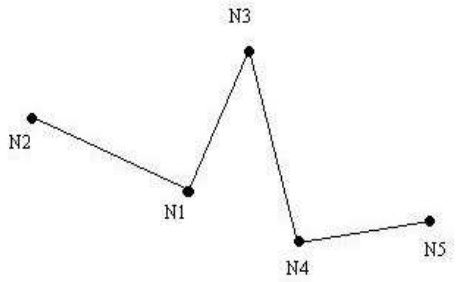

Fig. 1. Sensor network with 5 nodes

element occurs in $r = k$ blocks, every pair of elements occurs in λ blocks and every pair of blocks intersect in λ elements.
Example 5: Consider (v, k, λ) = (7,3,1), or equivalently (v, b, r, k, λ) = (7, 7, 3, 3, 1), Symmetric Design. Lets consider key-space, KS_1 = {1, 2, 3, 4, 5, 6, 7} having v = 7 objects. There are b = 7 blocks and each block contains k = 3 objects. Every object occurs in r = 3 blocks. Every pair of distinct objects occurs in λ = 1 blocks and every pair of blocks intersects in λ = 1 objects. The blocks of the Symmetric Design are:
{1, 2, 3}, {1, 4, 5}, {1, 6, 7}, {2, 4, 6}, {2, 5, 7}, {3, 4, 7}, {3, 5, 6}.

F. Latin square

A Latin Square on n symbols is an $n \times n$ arrangement such that each of the n symbols occurs exactly once in each row and in each column where n is the order of the square.
Example 6: A Latin square on 4 symbols, {1, 2, 3, 4}, is a 4×4 array shown below:

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \\ 3 & 4 & 1 & 2 \\ 4 & 3 & 2 & 1 \end{pmatrix}$$

G. Orthogonal Latin Squares

The Latin Squares $A = (a_{ij})$ and $B = (b_{ij})$, each of order n are *orthogonal* if all entries of the join of A and B are distinct.

H. MOLS

Latin Squares $A_1, A_2, ..., A_r$ are *Mutually Orthogonal Latin Squares* (MOLS) if they are orthogonal in pairs.

I. Complete Set

For prime power n, a set of $(n-1)$ MOLS of order n is called a Complete Set [10,11].
Example 7: For n = 4, there exists three MOLS each of order 4 that forms a complete set.

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \\ 3 & 4 & 1 & 2 \\ 4 & 3 & 2 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 2 & 1 \\ 2 & 1 & 4 & 3 \\ 3 & 4 & 1 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 4 & 1 & 2 \\ 4 & 3 & 2 & 1 \\ 2 & 1 & 4 & 3 \end{pmatrix}$$

J. Projective Plane

A Projective Plane is an abstract mathematical concept and can be defined as a set of lines and a set of points, the relation between the lines and points being called as incidence and satisfying the following properties:

- Given any two distinct points, there is exactly one line incident with both of them.
- Given any two distinct lines, there is exactly one point incident with both of them.
- There are four points such that no line is incident with more than two of them.

In other words, we can say that a projective plane of order $q > 1$ is such that;

- Any line is incident with $q + 1$ points.
- Any point is incident with $q + 1$ lines.
- There exists $q^2 + q + 1$ points and the same number of lines.

The smallest possible Projective plane is the Fano plane as shown in figure 2 . It has seven lines and seven points. It is a projective plane of order $q = 2$. Any line is incident with $2 + 1 = 3$ points. Any point is incident with $2 + 1 = 3$ lines. There exists $2^2 + 2 + 1 = 7$ points and an equal number of lines.

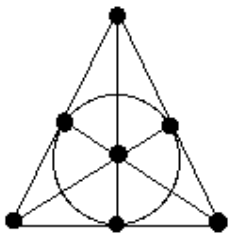

Fig. 2. Fano plane

III. Combinatorial based key generation

In the following sections, we describe the use of combinatorics in generation of key-chains for sensor nodes in sensor networks.

A. Mapping

We are interested in constructing Finite Projective Plane of order m which is nothing but a $(m^2 + m + 1, m + 1, 1)$ symmetric design.

We assume that the sensor nodes are deployed in large numbers randomly and the key-chains are stored in the ROMs of the nodes prior to deployment. Let N be the number of nodes in the network. Each sensor node has K keys in its key-chain. The K keys are selected from a Key-space KS which is a subset of the Key Pool P. Each node has to share λ keys to communicate securely.

If the sensor network contains N nodes, N number of key-chains has to be generated, i.e., a symmetric design with $b \geq N$ blocks needs to be generated from an object set S. We now partition the nodes into a k_n number of groups. Each node belonging to one group will pick the key-chain from the corresponding key-space. For example, if a node N_1 belong to group k_3, it will be deployed with a key-chain picked randomly from the 3^{rd} key-space KS_3. Hence k_n is also the number of key spaces generated for the network of size N.

Let m be a prime power such that there are $m^2 + m + 1$ nodes present in each group. Hence we construct a symmetric design with $v = b = m^2 + m + 1$ objects and blocks for the corresponding key-space. We can map the object set S of the symmetric design to the key pool P. A subset of the object set forms the key-space KS. The blocks b can be mapped to the key-chains K. Hence the number of blocks is equal to the number of key-chains, also equal to number of sensor nodes corresponding to a key-space. The objects in the block can be mapped to the keys in the key-chain. This provides $b \geq N$ key-chains each having $K = k = m^2 + m + 1$ keys in its key-chain. From the property of $(m^2 + m + 1, m + 1, 1)$ design, each pair of blocks can share $\lambda = 1$ object. This means, each pair which picked its key-chain from the same key-space has $\lambda = 1$ keys in common. The mapping is as shown in the Table 1.

TABLE I
Mapping from symmetric design to key generation.

Symmetric Design	**Key Generation**
Object Set (S)	Key Pool (P)
Subset of Object Set (KS)	Key-spaces (KS)
Subset size $(\|KS\| =$ v = $\mathrm{m}^2 + m + 1)$	Key-space size $\|KS\|$
Number of subsets (k_n)	Number of key-spaces (k_n)
Blocks	Key-chains
Number of blocks $(b = m^2 + m$ $+1)$	Number of Key-chains
Number of blocks $(b = m^2 + m$ $+1)$	Number of sensor nodes under each key-space
Number of objects in a block $(k = m + 1)$	Number of keys in key-chain
Number of blocks that an object is in $(r = m + 1)$	Number of key-chains that a key is in
Two blocks belonging to the same subset share $\lambda = 1$ objects	Two key-chains belonging to the same key-space share $\lambda = 1$ keys

B. Construction

To generate a $(m^2 + m + 1, m + 1, 1)$ design we use the following approach. First, we generate a complete set of $(m - 1)$ MOLS of order m. This can be used to construct a Affine plane of order m which makes it a $(m^2, m, 1)$ design. This can be converted to a projective plane which is a $(m^2+m+1, m+1, 1)$ design. The construction process can be summarized as follows:

1) Given N, the number of nodes in the network, find a prime power n where $n^2 + n + 1 \geq N$.
2) Find a suitable value for prime power m and the corresponding value of k_n.
3) Generate a complete set of $(m - 1)$ MOLS or order m [10, Appendix A]
4) Construct a Affine plane of order m from the MOLS [10]
5) Construct a Projective plane of order m by adding a point at infinity to the affine plane [10, Appendix B]
6) Repeat the steps 4 and 5 k_n times, i.e., for each key-space.

There are a number of ways in deciding the value of m and k_n. The approach we adopted was to find the value iteratively. We begin with the value of $m = n/2$ and compute the number of key-spaces necessary to form the required number of key-chains. That is, find k_n such that $k_n(m^2 + m + 1) \geq N$. Repeat the above process for prime powers less than m such that $k_n(m^2+m+1)-N$ is a feasible value. The value $k_n(m^2+m+1)-N$ is the number of extra key-chains generated by the design. Stop the iteration when this value becomes reasonably small. Since the iterations cannot be carried out indefinitely, we stop at four iterations and choose the value for m and k_n which gave the least value for $k_n(m^2 + m + 1) - N$.

The key-spaces are constructed such that any pair of key-spaces do not form a disjoint set. A pair of nodes which picked its key-chain from different key-spaces may or may not contain a key in common. Since, the key-spaces are non-disjoint, there exists a key-path of length greater than 1, if not a direct path.

Thus formed projective planes for each of the values k_n becomes the required key-chains.

IV. Analysis

In the modified symmetric design, any pair of blocks chosen from a single key-space has atleast one object in common ,i.e., any pair of key-chains chosen from a single key-space has atleast one key in common. The key-spaces are constructed such that two neighboring nodes whose key-chains are taken from different key-spaces have either a key in common or there exists a key-path between them. Hence, the probability of key share between any pair of nodes is 1. This implies that any two nodes can communicate securely.

Resilience can be defined as resistance against node capture. Compromise of security credentials, which are stored on

a sensor node or exchanged over radio links, should not reveal information about security of any other links in the network. Usually higher resilience means lower number of compromised links [12]. An attacker may be able to monitor the network and capture the nodes wisely or he may capture the nodes randomly. Say, an attacker tries to capture the nodes which have the same specific key in their key-chains. From the properties of symmetric design, for a single key space, we know that there are $m+1$ keys in the key-chain. Every pair of keys can occur in exactly one key-chain within that key space. Hence every $k_n(m^2+m)$ keys must be pairing with the specific key in these $k_n(m+1)$ key-chains. So, a wise attacker needs to capture $k_n(m+1)$ key-chains to compromise the entire network. But, if the attacker is unlucky and selects the nodes randomly. He might be capturing those $k_n(m^2)$ key-chains which do not pair with the specific key. Hence, he has to capture one more key-chain in each key-space to compromise the whole network. Therefore, an unlucky attacker will need to capture $k_n(m^2+1)$ key-chains to be able to recover all the keys or in other words, to compromise the entire network.

V. Results

With multiple key spaces the length of the key chain is $m+1$, which would have been $n+1$ in symmetric design which uses single key-space. Since $m < n/2$, the key chain length is very much reduced. This is a main advantage since the sensor nodes have limited memory. Even though the key chain length is reduced, the network remains connected with a high probability. The comparison of key-chain lengths for different number of nodes in the sensor network in case of both symmetric design with single key-space and symmteric design with multiple key-spaces is shown in figure 3.

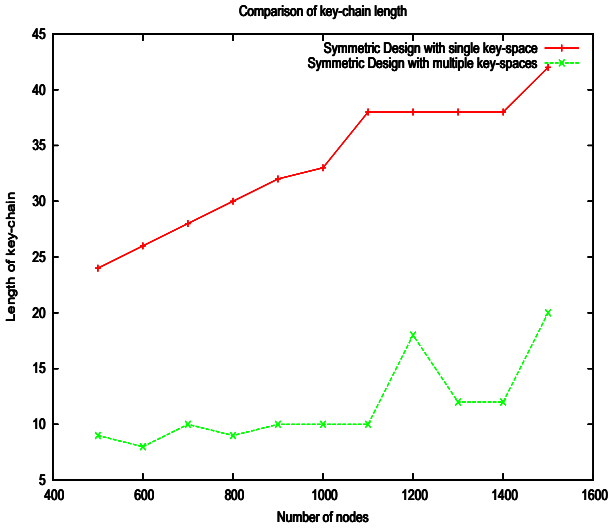

Fig. 3. Comparison of key-chain length

With the use of multiple key spaces, resilience of the network is not hampered. When a node is compromised the intruder has keys taken from one particular key-space. This information is not sufficient to comprise other parts of the network. Hence, he has to compromise more number of nodes to get enough keys to capture the entire network. The graph in figure 4 shows that the minimum number of nodes to be compromised is almost the same in symmetric design with single key-space and in symmetric design with multiple key-spaces.

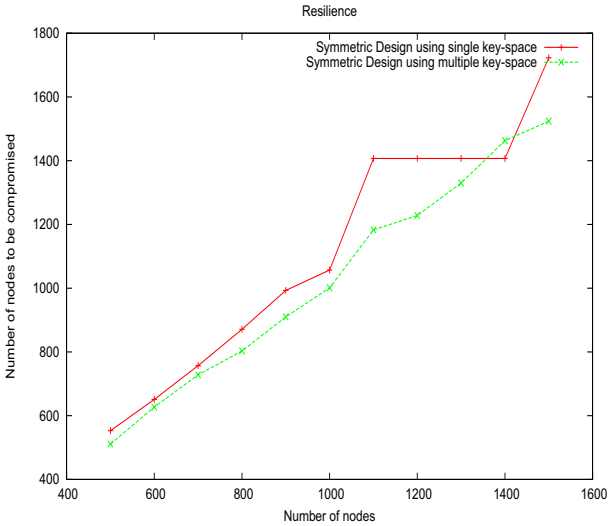

Fig. 4. Comparison of resilience

The combinatorial design generates more number of key-chains than the number of nodes in the sensor network. An efficient algorithm design should be such that the number of extra key-chains generated is as small as possible. This will ensure that unnecessary computations are avoided. The comparison of this computational overhead is shown in figure 5.

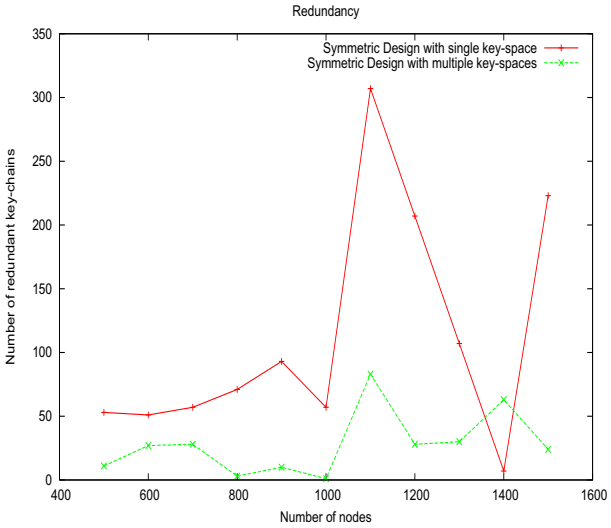

Fig. 5. Comparison of redundancy

VI. Conclusion

The novel idea brought in by this paper is the use of multiple key-spaces instead of a single key-space in the construction of key-chains. Adding to the advantage of using a deterministic approach, we have a few more results derived which prove that this method is better than using a single key-space. Generation of each key-space is left as a part of implementation. The more is the complexity involved in the key-space generation from the key-pool, the better this method proves out to be. The concept of $k_n(> 1)$ key-spaces itself proves lesser memory utilization, connectivity in the network and yet have the same network resilience. There exists no dependency between the number of keys in the key-chain and the number of nodes in the network.

References

1) B. C. Neumann, T. Tso., "Kerberos: An authentication service for computer networks", IEEE Communications 32(9):33-39, September 1994.
2) R. L. Rivest, A. Shamir and L. M. Adleman, "A method for obtaining digital signatures and public key cryptosystems", Communications of the ACM 21(2): 120-126, 1978.
3) W. Diffie and M. E. Hellman, "New directions in cryptography", IEEE Transactions in Information Theory 22: 644-654, November 1976.
4) L. Eschenauer, V. D. Gligor, "A key-management scheme for distributed sensor networks", Proceedings of the 9th ACM conference on Computer and Communications Security, 2002.
5) R. Merkle, "Secure Communication over Insecure Channels", Communications of the ACM, 1978.
6) H. Chan, A. Perrig and D. Song, "Random Key Predistribution Scheme for Sensor Networks", IEEE Symposium on Research in Security and Privacy, 2003.
7) W. Du, J. Deng, Y. S. Han and P. Varshney, "A Pairwise Key Pre-distribution scheme for Wireless Sensor Networks", Proceedings of the 10th ACM Conference on Computer and Communications Security, 2003.
8) J. Lee and D. R. Stinson, "Deterministic key predistribution schemes for distributed sensor networks", in Proceedings of ACM Symposium on Applied Computing , 2005.
9) Seyit A. Camtepe, Bulent Yener, "Combinatorial Design of Key Distribution Mechanisms for Wireless Sensor Networks", In IEEE/ACM Transactions on Networking, vol.15 no.2, April 2007.
10) I. Anderson, "Combinatorial Designs: Construction Methods", Ellis Horwood Limited, 1990.
11) C. J. Colbourn, J. H. Dinitz, "The CRC Handbook of Combinatorial Design", CRC Press, 1996.
12) Seyit A. Camtepe, Bulent Yener,"Key Distribution Mechanisms for Wireless Sensor Networks: a Survey", Technical Report TR-05-07 Rensselaer Polytechnic Institute, Computer Science Department, March 2005.

Appendix A

Theorem 1 A complete set of $(n-1)$ MOLS exists whenever n is prime.

Proof We define square arrays $A_1, A_2, ..., A_{n-1}$ as follows. For the (i,j)th entry of A_k, take $a_{ij}^{(k)} = ki + j$, reduced to modulo n to lie in the set 1,...,n

1) We first check that A_k is a Latin square. First, the entries in the ith row are all different; for if $a_{ij}^{(k)} = a_{iJ}^{(k)}$ then ki + j $\equiv ki + J \mathrm{mod}(n)$ whence $j = J$. Secondly, entries in the jth column are all distinct; for if $a_{ij}^{(k)} = a_{IJ}^{(k)}$ then $ki + jkI + jmod(n)$ whence $k(i-I) \equiv 0 \mathrm{mod}(p)$. Thus p divides $k(i-I)$, whence p divides $i-I$, so that $i \equiv I \mathrm{mod}(p)$ and i must equal I.
2) To check if A_k and A_h are orthogonal whenever $k \neq h$. Suppose $a_{ij}^{(k)} = a_{IJ}^{(k)}$ and $a_{ij}^{(h)} = a_{IJ}^{(h)}$. Then $ki + j \equiv kI + J$ and $hi + j \equiv hI + J \mathrm{mod}(n)$. Subtracting one from the other gives $(h-k)i \equiv (h-k)I \mathrm{mod}(n)$ and, as above this gives $i = I$. Substituting back now, this gives $j \equiv J \mathrm{mod}(n)$ whence $j = J$.

Appendix B

To constrcut a projective plane of order n given a complete set of MOLS of order n.

Steps Let there be a given a complete set of MOLS of order n. Then we can create a $n^2 \times (n+1)$ matrix (a_{ij}) as follows: The first two columns of the matrix form a succession of ordered pairs, namely, (0,0), (0,1),...,(0,n-1), (1,0), (1,1),..., (1,n-1),...,(n-1,0), (n-1,1),...,(n-1,n-1). The third column repeats the entries of the Latin square A_1 by reading it row by row from left to right. The fourth column does the same for Latin square A_2 and so on.

Label the n^2 row-vectors of (a_{ij}) as 'points' of our projective plane. Create $(n+1)$ 'ideal points' and label them $z_1, z_2, ..., z_{n+1}$. Altogether we now have $n^2 + n + 1$ points.

The 'lines' of the plane are stipulated as follows. First, there is an 'ideal line', called L, defined as the set $\{z_1, z_2, z_3, ..., z_{n+1}\}$. The remaining $n^2 + n$ lines (called 'ordinary lines') are labeled ij, where i = 0, 1, 2,..,$(n-1)$, j = 1, 2,...,$(n+1)$. Each such line ij is the set

$\{z_j$, those 'points' (= row-vectors) which have i in their jth column$\}$.